(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number : **0 528 584 A1**

(12) **EUROPEAN PATENT APPLICATION**

(21) Application number : **92307154.2**

(51) Int. Cl.^{5} : **G06F 12/10**

(22) Date of filing : **05.08.92**

(30) Priority : **12.08.91 US 744203**

(43) Date of publication of application : **24.02.93 Bulletin 93/08**

(84) Designated Contracting States : **DE FR GB**

(71) Applicant : **International Business Machines Corporation**
Old Orchard Road
Armonk, N.Y. 10504 (US)

(72) Inventor : **Chiarot, Kevin Arthur**
211 Mansion Street
Poughkeepsie, New York 12601 (US)
Inventor : **Schmalz, Richard John**
7 Edge Hill Drive
Wappinger Falls, New York 12590 (US)
Inventor : **Schmitt, Theodore Jerome**
98 Clifton Avenue
Kingston, New York 12401 (US)
Inventor : **Tran, Arnold Steven**
6 Mary Lou Lane
Shokan, New York 12481 (US)
Inventor : **Tung, Shih-Hsiung Stephen**
18 Lainey Lane
Kingston, New York 12401 (US)

(74) Representative : **Atchley, Martin John Waldegrave**
IBM United Kingdom Limited Intellectual Property Department Hursley Park
Winchester Hampshire SO21 2JN (GB)

(54) **Directory look-aside table for a virtual data storage system.**

(57) The present invention relates to a directory look-aside table (65) for a virtual data storage system which can have pages in different address spaces having the same virtual addresses, the addresses (60) being identified by respective segment tables.

According to the invention the table is characterised in that it comprises first means (61) for distinguishing between address spaces and data spaces, second means (62) for subclassifying data spaces by one or more space identification discriminating low order bits, and third means (63, 64) responsive to said first and second means for providing a unique algorithm for each sub-class of data spaces for the assignment of virtual page address to directory look-aside table rows using virtual page-address bits.

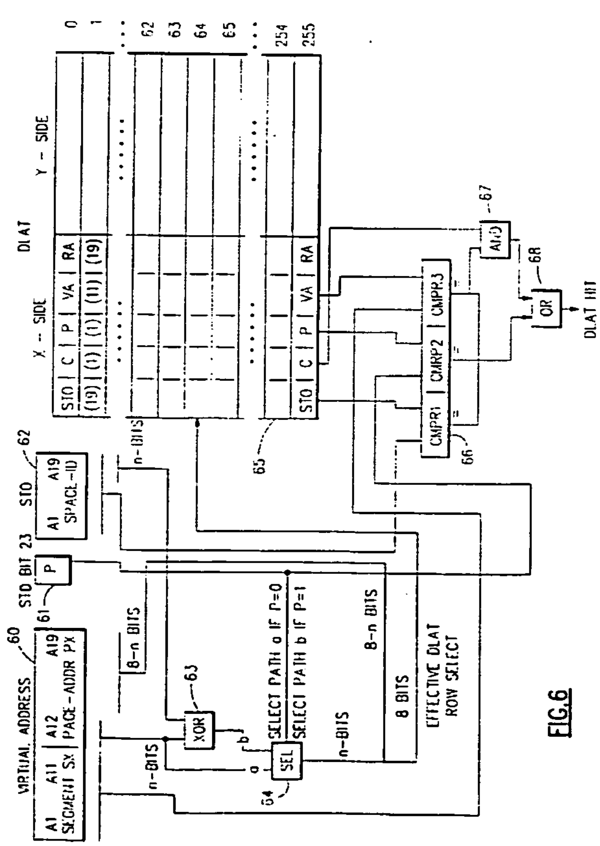

FIG.6

EP 0 528 584 A1

The present invention generally relates to virtual data storage mechanisms for data processing systems and, more particularly, to a directory look-aside table (DLAT) which is capable of improved handling of the periodicity of virtual page addresses when processing among multiple data spaces. The invention is specifically directed to the minimisation of synonym entries in the DLAT for a system having DLAT entries that can concurrently translate virtual addresses in a plurality of data spaces into real main storage addresses.

Description of the Prior Art

Virtual storage organisation and management for data processing systems are described, for example, by Harvey M. Deitel in "An Introduction to Operating Systems", Addison-Wesley (1984), by Harold Lorin and Harvey M. Deitel in "Operating Systems", Addison-Wesley (1981), and by Harold S. Stone in "High-Performance Computer Architecture", Addison-Wesley (1987). In a virtual storage system, paging is a relocation and address-to-physical -location binding mechanism providing the user of the system with what appear to be considerably larger memory spaces than are really available. The key feature of the virtual storage concept is disassociating the addresses referenced in a running process from the addresses available in main storage. The addresses referenced by the running process are called virtual addresses, while the addresses available in main storage are called real addresses. The virtual addresses must be mapped into real addresses as the process executes, and that is the function of the dynamic address translation (DAT) mechanism. One such mechanism employs a directory look-aside table (DLAT), sometimes referred to as a translation look-aside buffer (TLB), which stores recent virtual address translations. For virtual addresses stored in the DLAT, the translation process requires only a single or, at most, a couple of machine cycles. For addresses not stored in the DLAT, the DAT process may take from fifteen to sixty cycles.

Translations from the virtual address to the real address must be made to find where the addressed instruction or data is in main storage. This is typically done on a page basis. In fact, the translations stored in the DLAT are actually only page translations, and the last bits of an address are the location in that page, so only the page address must be translated.

In conventional virtual storage systems, a condition called thrashing can occur wherein the system can do little useful work because of excessive paging. The condition was recognised and discussed, for example, by P. J. Denning in "Thrashing: its Causes and Prevention", AFIPS Conf. Proc., vol. 33, 1968 FJCC, pp. 915-922. Denning maintained that for a program to run efficiently, its working set of pages must be maintained in primary storage; otherwise, thrashing will occur as the program repeatedly requests pages from secondary storage. The condition is mentioned in Deitel, supra, in section 9.5, "Working Sets" in his chapter on Virtual Storage Management.

US -A- 4,136,385 addresses the problem with a synonym control for multiple virtual storage systems. The US -A- 4,136,385 DLAT synonym control controls the setting of an indicator in each DLAT entry for indicating whether the DLAT entry is to be shared by all user address spaces or is to be restricted to a single address space identified in the DLAT. This is accomplished by means of a common space bit in any segment table entry (STE) or, alternatively, in any page table entry (PTE) in any private address space to indicate whether the segment or page contains programs and data private to the address space or shared by all address spaces. Thus, each DLAT entry contains a common/private storage indicator which is set to the state of the common space bit in the STE or PTE used in an address translation loaded into the DLAT entry. When the entry is read, the private/common storage indicator controls whether the DLAT can only be used by the address space identified in the DLAT or by all address spaces.

Conventional two-way DLAT designs inadequately handle the periodicity of virtual page addresses when processing among a plurality of data spaces.

A typical two-way DLAT design maps identical virtual addresses in all data spaces to the same DLAT row. For example, in the following pseudocode

```
DO i = 1 to 1,000
A(i) + B(i) = C(i),
```

if A, B and C have the same origin (e.g., zero) in each of three data spaces, the code will deliver poor performance due to DLAT thrashing caused by data space synonyms. While the US -A- 4,136,385 synonym control eliminates a class of synonym entries by means of a common space bit, it does not address the class of synonym problem in a two-way DLAT produced by three data spaces having the same origin.

The object of the present invention is to provide an improved directory look-aside table for a virtual data storage system.

The present invention relates to a directory look-aside table for a virtual data storage system which can have pages in different address spaces having the same virtual addresses, the addresses being identified by respective segment tables.

According to the invention the table is characterised in that it comprises first means for distinguishing between address spaces and data spaces, second means for sub-classifying data spaces by one or more space identification discriminating low order bits, and third means responsive to said first and second means for providing a unique algorithm for each subclass of data spaces for the assignment of virtual page address to directory look-aside table rows using

virtual page-address bits.

According to one embodiment the invention, there is provided an improved DLAT logic which uses a "private space bit", corresponding to the common space bit described in US -A- 4,136,385 synonym control, to select different DLAT addressing algorithms. Data spaces are further sub-classified using space identification bits, and for each sub-class, a unique algorithm is selected based on the page address bits. An Exclusive OR function is used to generate the DLAT selection bits.

This approach minimises private space synonyms while maximising common space synonyms. The result is improved performance since the former minimises thrashing and the latter maximises the value of the DLAT common bit.

In order that the invention may be more readily understood an embodiment will now be described with reference to the accompanying drawings, in which:

Figure 1 is a block diagram illustrating the format of a known form of virtual address used in a data storage system,

Figure 2 is a block diagram of a conventional dynamic address translation structure,

Figure 3 is a block diagram illustrating the format of a Segment Table Designation (STD) showing the private-space bit,

Figure 4 is a fragment of Figure 3 showing Segment Table Origin (STO) bits used as SPACE-ID bits,

Figure 5 is a table showing an Exclusive OR operation in accordance with the invention using a STD bit to define two sub-classes,

Figure 6 is a block diagram showing an arrangement in accordance with the invention implementing the Exclusive OR operation shown in Figure 5,

Figure 7 is a table showing the Exclusive OR operation using two space identification bits to define four sub-classes; and

Figure 8 is a table showing the Exclusive OR operation using three space identification bits to define eight sub-classes.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT OF THE INVENTION

The description which follows uses the term "DLAT" for a directory look-aside table, but those skilled in the art will understand that this term may be used interchangeably with "TLB" for translation look-aside buffer. For purposes of the following description, a paging/segmentation virtual address system is assumed. In such systems, the virtual address format is as shown in Figure 1 and comprises s-bits for the segment index (SX), p-bits for the page index (PX), and d-bits for the displacement index (DX). The virtual address may be, for example, 31 bits of which bits A1 through A11 comprise the segment bits, bits A12 through A19 comprise the page bits, and bits A20 through A31 comprise the displacement bits.

As shown in the known arrangement in Figure 2, the virtual address is generated by address generator 20. The address generator 20 is part of a central data processing unit (CPU) (not shown). The most recently referenced pages have entries in the DLAT 21. For a DLAT with 256 congruence classes, bits A12 through A19 of the virtual address are used to address the DLAT. The virtual page identification bits from the addressed entry read out of the DLAT 21 are compared in comparator 22 with bits A1 through A11 of the virtual address.

If there is no match, a DLAT miss has occurred. When a DLAT miss occurs, address translation is obtained through, for example, a segment/page table search and placed in the DLAT. The stncture which performs this search is referred to as the buffer control element (BCE). The segment/page table search begins by adding the value in the segment table origin register 23 and the bits A1 to A11 of the virtual address in adder 24 to obtain an index value for the segment map table 25. The entry output from the segment map table 25 is, in turn, used as an index for the page map table 26 entry, there being a separate page map table for each segment. The entry output from the page map table 26 provides the page frame at which the virtual page resides in real storage and is passed by OR gates 27 and concatenated with the displacement bits A20 through A31 of the virtual address generator 20 to form the real address in real address register 28.

On the other hand, if there is a match in the DLAT 21, the comparator 22 enables AND gates 29 which pass the entry output from the DLAT 21 to OR gates 27. In this case, the entry output from the DLAT 21 is the associated real address field which is concatenated to the displacement bits A20 through A31 to form the real storage address in register 28. Obviously, this process of address translation is considerably faster than that of the segment/page table search which occurs when a DLAT miss occurs. The segment/page table search may take fifteen to eighty cycles to complete, whereas a DLAT access can be completed in one cycle.

Normally, most address translation requests are made by a search of the DLAT, and, while the segment/page table search takes a greater number of processor cycles than making the translation by means of the DLAT, the segment/page table search is itself not without the possibility of a translation failure. For example, the segment map table search may indicate that the segment is not in primary or main storage, causing the operating system to locate the segment in secondary storage, e.g., a direct access storage device (DASD), create a page table for the

segment, and load the appropriate page into primary storage, possibly replacing an existing page in the process.

Even if the segment is in primary storage, the desired page may not be in primary storage, causing the operating system to locate the page in secondary storage and to load the page into primary storage, again possibly replacing an existing page in the process. The process of accessing secondary storage can take up to several hundred processor cycles.

The foregoing description is for a conventional prior art DLAT structure. A problem with this conventional DLAT stncture is that it does not handle the problem of synonym entries in the DLAT in data processing systems which can simultaneously translate virtual addresses for a plurality of address spaces into real main storage addresses wherein different address spaces must have access to the same set of shared programs and data. That problem was addressed in US -A- 4,136,385, supra, which provided special controls employing a common space bit. This, however, does not address the problem caused by the periodicity of virtual page addresses when processing among a plurality of data spaces. That is, for the periodic processing example of

$$A(i) + B(i) = C(i),$$

where A, B and C have the same virtual page addresses (e.g., zero) but with different data spaces, with a conventional two-way (X/Y sides) DLAT stncture, the associated real addresses for A(i) and B(i) will be loaded into the X and Y sides of the DLAT respectively at address zero after address translation is completed. The associated real address for C(i) will be loaded into the X side of the DLAT at the same address zero after address translation, and that will overwrite the DLAT entry for A(i). For the next loop, A(i) will miss the DLAT compare and take many cycles to be re-translated. Then the associated real address will be loaded into the Y side of the DLAT, and that will overwrite the DLAT entry for B(i). This kind of DLAT thrashing will cause dramatic system performance degradation.

The arrangement to be described herein distinguishes between address spaces and data spaces. This kind of identification mechanism is known, as described in US -A- 4,136,385, supra. Bit "23" of the Segment Table Designation (STD) is defined as a private-space (data-space) bit. Figure 3 shows the STD-bit "23" labelled with the letter "p". This STD-bit, when a logical "1", is used to obtain the separation of data spaces in a DLAT. The STD-bit "23", when a logical "0", is used to retain the DLAT performance value of the common-segment bit for address spaces.

The arrangement to be described builds on this mechanism by sub-classifying data spaces by one or more space identification (SPACE-ID) discriminating (e.g., non-constant or random) low-order bits. Figure 4 shows the Segment Table Origin (STO) bits "1" to "19" as SPACE-ID bits. In the International Business Machine Corporation (IBM) MVS system, these bits are stored in the STO register and may be unhashed non-constant, low-order bits or the low-order output bits of a hash of many STO non-constant bits. For each sub-class of data spaces, virtual page address (PAGE-ADDR) bits are used to select a unique algorithm for the assignment of virtual page addresses to the DLAT rows. These bits may be unhashed PAGE-ADDR bits or output bits of a hash of many PAGE-ADDR hits. The MVS environment refers to IBM Multiple Virtual Storage (MVS) operating system introduced in mid-1974 for their line of main frame computers. For a general discussion of this operating system, see for example Chapter 21 of Deitel in "An Introduction to Operating Systems", supra.

In an exemplary embodiment, a two data-space sub-class design is realised by an Exclusive OR of a data space unhashed STO bit, such as bit "17" in an MVS environment, and the virtual page unhashed PAGE-ADDR high-order bit. The consequences of this design are as follows. For even sub-class data spaces (STO bit "17" = 0), data space page addresses with a 0-value PAGE-ADDR high-order bit are assigned to even rows of the DLAT, and data space page addresses with a 1-value PAGE-ADDR high-order bit are assigned to odd rows of the DLAT. For odd sub-class data spaces (STO bit "17" = 1), data space page addresses with a 0-value PAGE-ADDR high-order bit are assigned to odd rows of the DLAT, and data space page addresses with a 1-value PAGE-ADDR high-order bit are assigned to even rows of the DLAT.

Given a 256x2 DLAT, if the spaces A and C are in sub-class zero and the space B is in sub-class one, then each corresponding even page of A is assigned to a DLAT even row, each corresponding even page of B is assigned to a DLAT odd row, and each corresponding even page of C is assigned to a DLAT even row. In like manner, each corresponding odd page of A is assigned to a DLAT odd row, each corresponding odd page of B is assigned to a DLAT even row, and each corresponding odd page of C is assigned to a DLAT odd row.

Figure 5 shows a tabular representation of the Exclusive OR operation and illustrates the foregoing relation. Note that in Figure 5 the zero sub-class data space assignment rule is the same as the assignment rule for address spaces. For the periodic processing example of A(i) + B(i) = C(i), if at least one of the three data spaces is in a different sub-class from the other two data spaces as determined by STO low-order bits, there will be little or no thrashing of a two-way DLAT. This separation of data spaces is due to the fact that MVS assignment of STO bits to data spaces is usually random. In fact the probability of all three spaces referenced being in the same sub-class is .25. Moreover, when STO bits are assigned in se-

quence by MVS, the probability of all three consecutively created data spaces being in the same subclass is zero.

The logic for DLAT handling according to a preferred embodiment of the invention is shown in Figure 6, to which reference is now made. The logic receives as inputs bits from the virtual address 60, the STD bit "23" denoted by reference numeral 61, and SPACE-ID bits from the Segment Table Origin (STO) bits 62. Exclusive OR gate 63 receives n bits of the PAGE-ADDR of the virtual address 60 and n bits from the SPACE-ID 62. The n-bits of PAGE-ADDR are supplied as input "a" to the selector 64, while the output of Exclusive OR gate 63 is supplied as input "b" to selector 64. The operation of the selector 64 is controlled by the "p" bit 61.

The general case of n data-space sub-classes is obtained by generating a unique set of n bits for each combination of n discriminating low-order SPACE-ID bits and n high-order virtual page address (PAGE-ADDR) bits. The process of generation of the n high-order effective row select bits for the DLAT 65 is from selector 64 and Exclusive OR 63. The DLAT 65 structure is conventional, the preferred structure being that which is implemented in IBM 3090-S machines. Each DLAT entry has fields for containing nineteen STO bits, one common space bit, one private space bit, eleven high-order virtual address bits, and nineteen high-order real address bits.

When "p" bit 61 is zero, the selector 64 will select the "a" input n high-order PAGE-ADDR bits 60. When the "p" bit 61 is one, the selector 64 will select the "b" input n bits generated by Exclusive OR 63 operating on n discriminating low-order SPACE-ID (STO) 62 bits with n high-order PAGE-ADDR 60 bits. The 8-n low-order effective DLAT row select bits for the DLAT 65 are the unhashed 8-n low-order page address bits from the page address field, PX, of the virtual address 60. Note that if the Exclusive OR function is not performed but low-order SPACE-ID bits 63 are substituted unchanged, the result is a partitioning of the DLAT rather than a re-allocation of the total DLAT.

A DLAT hit is determined by a three-way compare function performed by comparator 66. The CMPR1 function generates a binary "1" when the SPACE-ID bits 62 and the STO bits read out from the DLAT 65 are equal. The CMPR2 function generates a binary "1" when the "p" bit 61 and the corresponding "p" bit read out from the DLAT 65 are equal. The CMPR3 function generates a binary "1" when the segment address bits of the virtual address 60 and the virtual address (VA) read out from the DLAT 65 are equal. A DLAT hit is indicated when all three compare functions generate a binary "1" output. A DLAT hit can also be made when the common space "c" bit of the selected DLAT entry is a binary "1" and the CMPR3 function generates a binary "1". The architecture prevents the case of the "p" bit and the "c" bit both being "1s" for the same address translation; therefore, any DLAT entry would not have the "p" and "c" bits both binary "1s".

For an implementation where n equals two, the resultant values of this Exclusive OR operation, for all combinations of two SPACE-ID bits and two high-order PAGE-ADDR bits, are shown in the table of Figure 7. Note that the zero sub-class data space assignment rule is the same as the assignment rule for address spaces.

With the "n equals two" implementation, the DLAT is logically divided into four sub-classes for the virtual PAGE-ADDR bits. Given a 256x2 DLAT with n equal two and if spaces A, B and C being in sub-classes "00", "01" and "10", respectively, then for the periodic processing example of A(i) + B(i) = C(i), there will be no thrashing of a two-way DLAT. Since MVS assigns STD bits to data spaces in a random fashion, the probability of all three data spaces referenced being in the same sub-class is small (e.g., .0625). When the STD bits are assigned in sequence, which may occur with MVS, the probability of all three consecutively created data spaces being in the same subclass is zero.

For an implementation where n equals three, the resultant values of the Exclusive OR operation for all combinations of three SPACE-ID bits and three high-order PAGE-ADDR bits are shown in the table of Figure 8. Mote again that the zero sub-class data space assignment rule is the same as the assignment rule for address spaces. The practical effect with the "n equals three" implementation is that the DLAT is logically divided into eight sub-classes for the same virtual PAGE-ADDR. Thus, for the periodic processing example of A(i) + B(i) = C(i), there will be almost no thrashing of a two-way DLAT. This minimum thrashing follows from MVS assignment of STD bits to data spaces in a random fashion. The probability of all three spaces referenced being in the same sub-class is very small (i.e., .0156).

For an implementation where n equals eight, there is maximum dispersion of assignment across all data space DLAT sub-classes for like PAGE-ADDR bits of data spaces. This dispersion minimises DLAT thrashing due to data space synonyms. On the other hand, there is, effectively, maximum concentration of assignment to the zero data space DLAT sub-class for like PAGE-ADDRs of address spaces. This concentration allows the performance advantage of common-segment bit for address spaces.

Claims

1. A directory look-aside table (65) for a virtual data storage system which can have pages in different address spaces having the same virtual addresses, the addresses (60) being identified by re-

spective segment tables, characterised in that it comprises

first means (61) for distinguishing between address spaces and data spaces,

second means (62) for sub-classifying data spaces by one or more space identification discriminating low order bits, and

third means (63, 64) responsive to said first and second means for providing a unique algorithm for each sub-class of data spaces for the assignment of virtual page address to directory look-aside table rows using virtual page-address bits.

2. A directory look-aside table as claimed in claim 1 characterised in that said first means (61) generates a private-space bit (p) which, when set, designates a data space.

3. A directory look-aside table as claimed in claim 2 characterised in that said space identification discriminating low order bits are n segment table bits where n defines the number of sub-classes of said data spaces.

4. A directory look-aside table as claimed in claim 3 characterised in that the number of said space identification discriminating low order bits is two.

5. A directory look-aside table as claimed in claim 3 characterised in that the number of said space identification discriminating low order bits is three.

6. A directory look-aside table as claimed in claim 3 characterised in that said third means comprises

Exclusive OR means (63) for combining n-bits of a page address of a virtual address with n-bits of said segment table origin bits to produce a first output; and

selector means (64) responsive to said first means for selecting said first output for form a directory look-aside table row select signal when said private-space bit is set but, otherwise, selecting said n-bits of said page address to form said directory look-aside table row select signal.

7. A directory look-aside table as claimed in claim 6 further characterised in that it comprises three-way compare means (66) responsive to said first and second means, a segment portion of said virtual address and an output from said directory look-aside table for determining a hit for a real address read out of said directory look-aside table.

8. A directory look-aside table as claimed in claim 7 characterised in that said three-way compare means comprises

first compare means (CMPR1) for comparing said segment table origin bits with corresponding segment table origin bits read out of said directory look-aside table;

second compare means (CMPR2) for comparing said private-space bit with a corresponding private-space bit read out of said directory look-aside table; and

third compare means (CMPR3) for comparing said segment portion of said virtual address with a corresponding portion of a virtual address read out of said directory look-aside table, a compare by all three of said first, second and third compare means indicating a hit.

9. A directory look-aside table as claimed in claim 8 further characterised in that it comprises means for distinguishing between common address space and private address space, a common-space bit when set designating common address space, setting of said common-space bit and said private-space bit being mutually exclusive, a hit also being determined by a compare by said third compare means and said common-space bit being set.

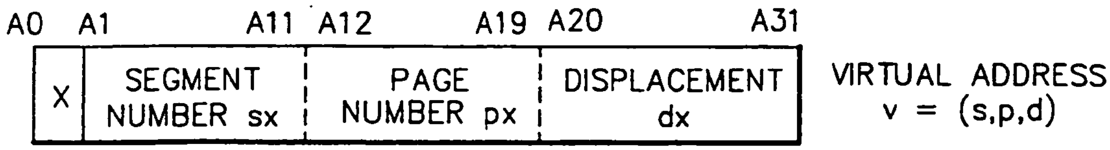

FIG.1
Prior Art

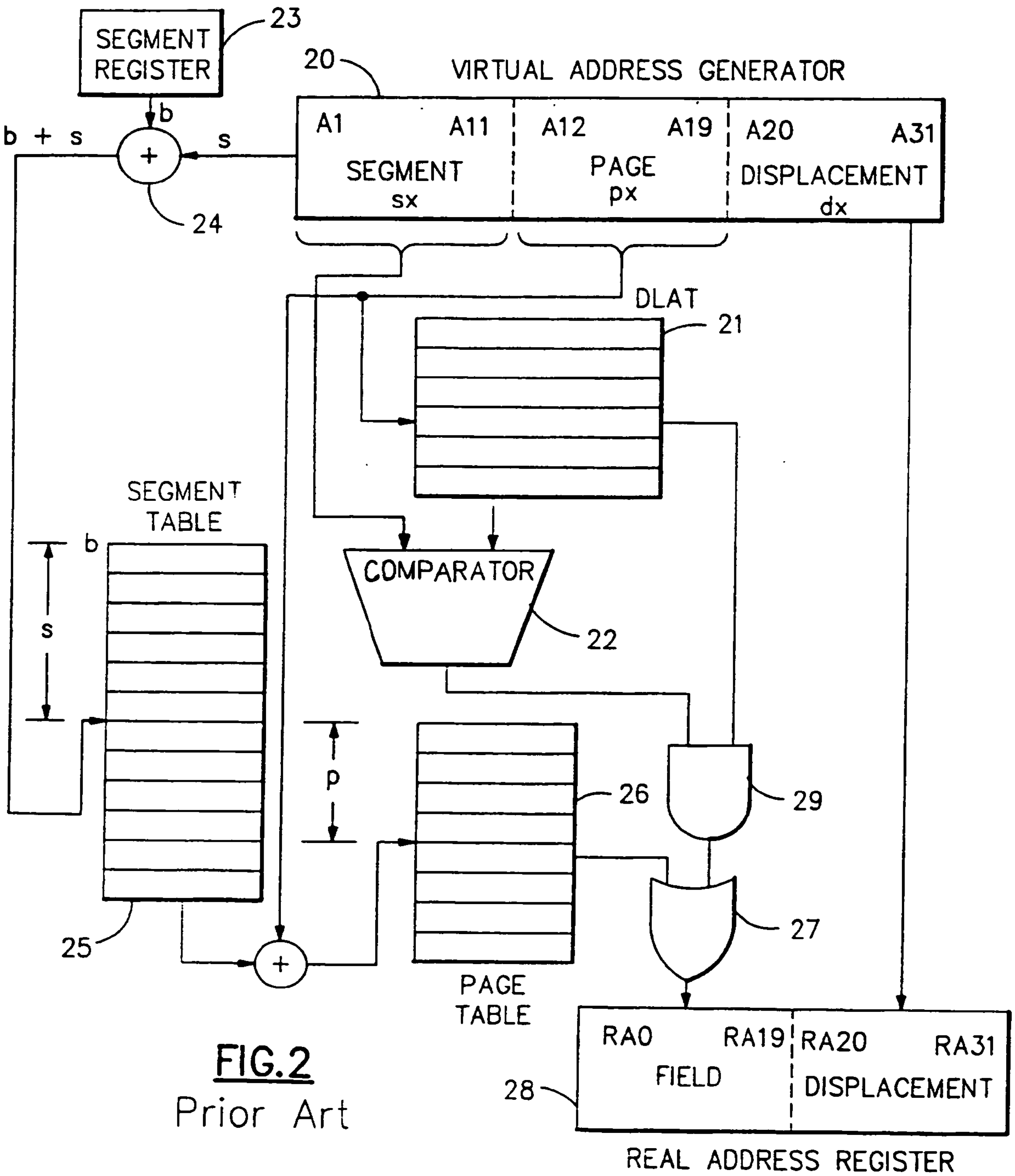

FIG.2
Prior Art

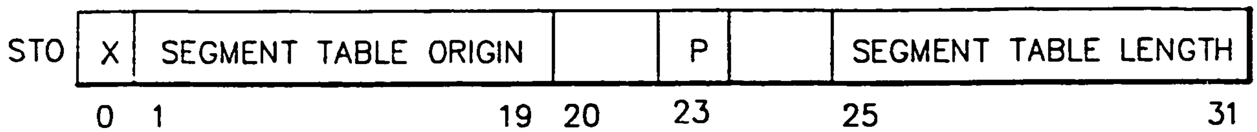

FIG.3

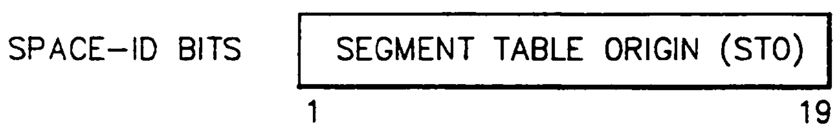

FIG.4

(STO) SPACE-ID BITS SUB-CLASSES	PAGE-ADDR HIGH ORDER BITS 0	1
0	0	1
1	1	0

EFFECTIVE DLAT ROW SELECT HIGH ORDER BIT

FIG.5

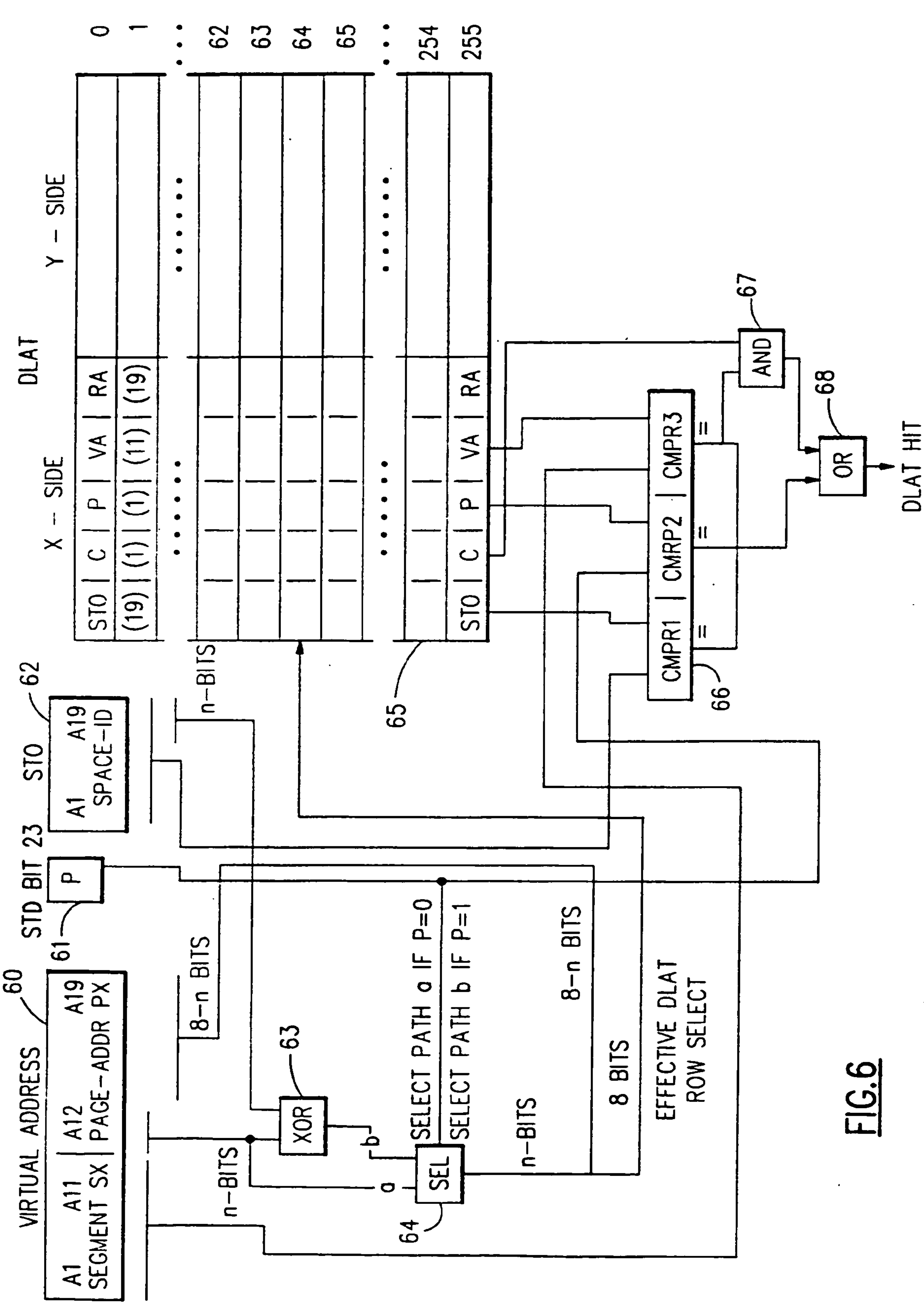
VIRTUAL ADDRESS
60
A1 SEGMENT SX
A11
A12 PAGE-ADDR PX
A19
STD BIT 23
61
P
STO
62
A1 SPACE-ID A19
n-BITS
8-n BITS
63
XOR
b
a
SEL
64
SELECT PATH a IF P=0
SELECT PATH b IF P=1
n-BITS
8-n BITS
8 BITS
EFFECTIVE DLAT ROW SELECT
DLAT
X - SIDE
Y - SIDE
STO | C | P | VA | RA
(19) | (1) | (1) | (11) | (19)
STO | C | P | VA | RA
0
1
62
63
64
65
254
255
65
CMPR1
CMRP2
CMPR3
=
66
AND
67
OR
68
DLAT HIT

FIG.6

(STO) SPACE-ID BITS SUB-CLASSES	PAGE-ADDR HIGH ORDER BITS 00	01	10	11
00	00	01	10	11
01	01	00	11	10
10	10	11	00	01
11	11	10	01	00

EFFECTIVE DLAT ROW SELECT HIGH ORDER BITS

FIG.7

(STO) SPACE-ID BITS SUB-CLASSES	PAGE-ADDR HIGH ORDER BITS 000	001	010	011	100	101	110	111
000	000	001	010	011	100	101	110	111
001	001	000	011	010	101	100	111	110
010	010	011	000	001	110	111	100	101
011	011	010	001	000	111	110	101	100
100	100	101	110	111	000	001	010	011
101	101	100	111	110	001	000	011	010
110	110	111	100	101	010	011	000	001
111	111	110	101	100	011	010	001	000

FIG.8

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 92 30 7154

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int. Cl.5)
X Y	US-A-4 774 659 (SMITH ET AL.) * column 18, line 3 - line 34; figure 7 * * column 15, line 47 - column 17, line 21 *	1-6 7-9	G06F12/10
X	PATENT ABSTRACTS OF JAPAN vol. 005, no. 107 (P-070)11 July 1981 & JP-A-56 047 980 (NEC CORP.) 30 April 1981 * abstract *	1-6	
Y	DE-A-4 030 287 (HITACHI) * column 4, line 13 - column 5, line 50; figure 1 *	7-9	
Y	DE-A-3 633 227 (SIEMENS) * abstract; figure 5 *	1-6	
Y	DE-A-4 019 961 (HITACHI) * column 3, line 66 - column 4, line 44; figures 1-3 *	1-6	TECHNICAL FIELDS SEARCHED (Int. Cl.5)
A	EP-A-0 327 798 (INTERNATIONAL BUSINESS MACHINES) * column 4, line 54 - column 6, line 53; figures 1,2,6 *	1-9	G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	20 NOVEMBER 1992	NIELSEN O.P.

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document